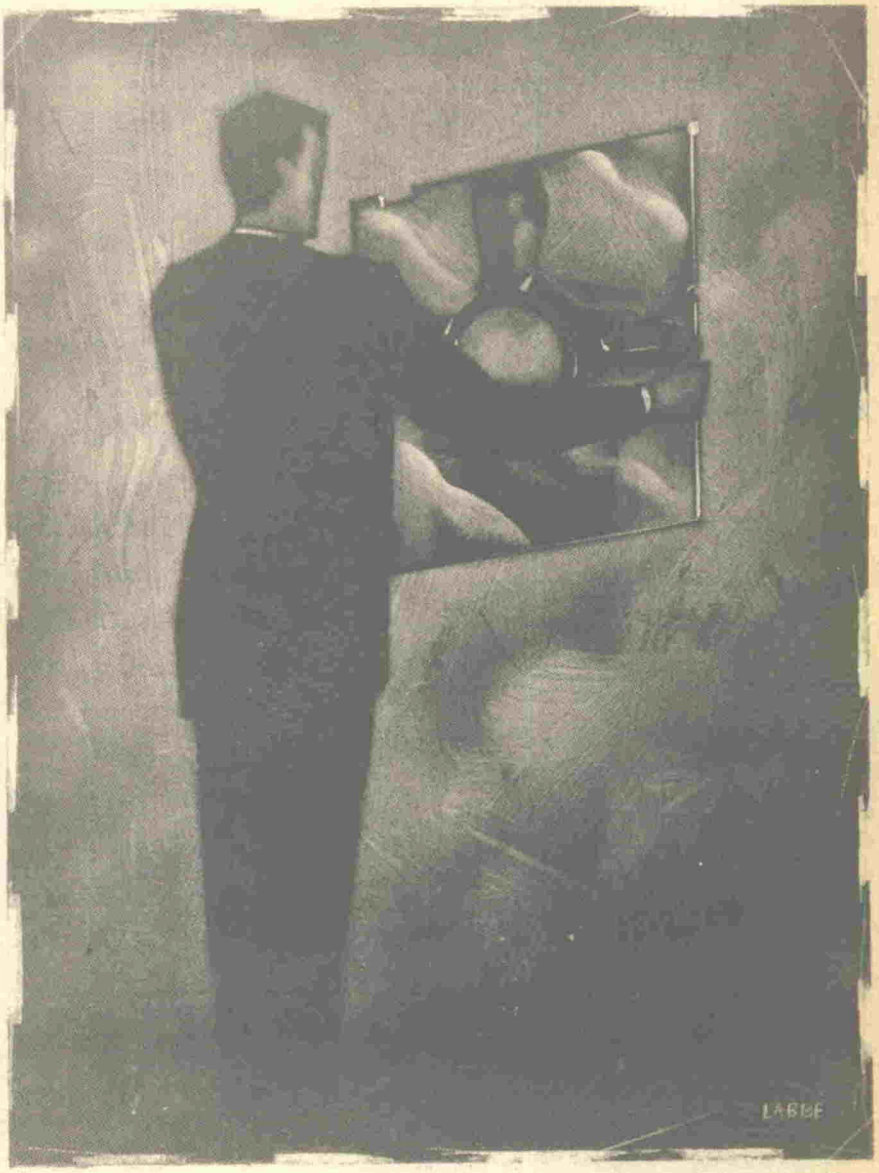
ดร.ปัญญา เปรมปรีดิ์

เทคนิคน่ารู้การเขียนโปรแกรมภาษาปาสคาล (2)

พบกับงานเขียนอย่างมีหลักเกณฑ์ในการเขียนโปรแกรมการใช้สัปรูทีนระดับเดียวกันและสัปรูทีนใหญ่ ๆ เพราะเป็นเทคนิคที่น่าเรียนรู้



ในครั้งที่แล้วผมได้พูดถึงกฎเกณฑ์ "Scope Rule" วิธีการวิ่งของโปรแกรมในภาษาปาสคาลและซี และได้พูดถึงการแบ่งชนิดของตัวแปร (variable) ออกเป็น 4 ชนิด คือ โกลบอล, โลคอล อินพุตพารามิเตอร์ และเอาต์-พุตพารามิเตอร์ และได้ให้ข้อเสนอแนะไว้ 6-7 ข้อ แต่ผมไม่มีโอกาสลงในรายละเอียดของข้อแนะนำ 2 ข้อหลัง ผมจึงอยากจะเอามาอธิบายกันในวันนี้

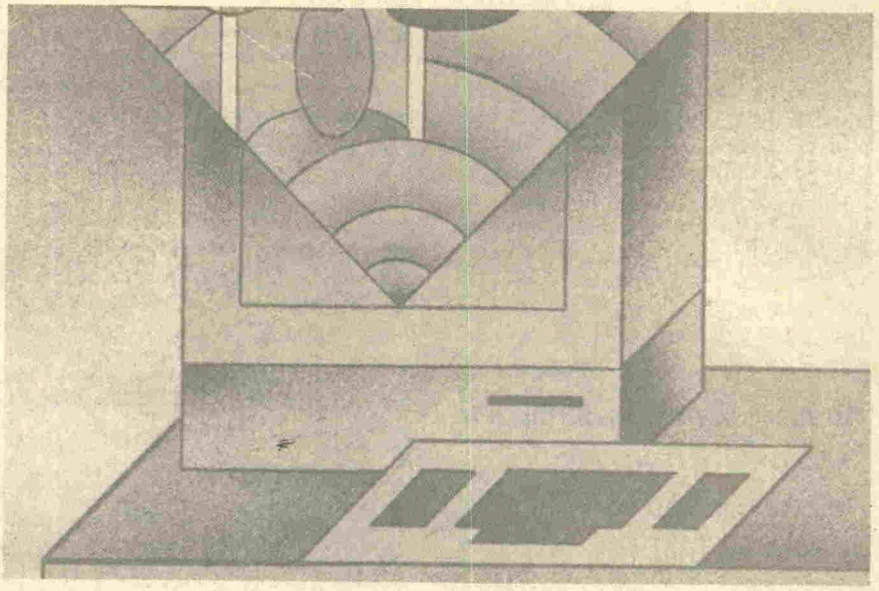
ทบทวนความรู้

ก่อนจะเข้าไปในรายละเอียด ผมใคร่ขอทบทวนความรู้เสียหน่อยหนึ่งคือภาษาปาสคาลและซีนี้จะมีลักษณะพิเศษอยู่หนึ่งอย่างคือ ให้เขียนรายละเอียดของสัปรูทีน (เรียกว่า "PROCEDURE" ซึ่งผมอ่านว่า "โพรซีเดอร์" ตามแบบคนอังกฤษ) เอาไว้ก่อน เสร็จแล้วจึงจะถูกเรียกใช้ในบอดี้ของเมนโปรแกรมหรือในสัปรูทีนอื่น ๆ การเรียกใช้กันและกันนี้มีกฎเกณฑ์ว่า สัปรูทีนที่อยู่ในระดับข้างเคียงกันนั้น ตัวที่เขียนอยู่ด้านล่างจะเรียกใช้ตัวที่อยู่ข้างบนได้ แต่ขณะเดียวกัน สัปรูทีนอาจมีสัปรูทีนย่อยอยู่ภายในก็ได้ ดังนั้นการเขียนโปรแกรมของเรานั้นถ้าจะต้องมีการทำงานแบบเรียกใช้สัปรูทีนซ้อน ๆ กันลงไปก็จะทำได้ 2 แบบ คือ 1. ให้เขียนสัปรูทีนใหญ่เอาไว้ด้านล่าง สัปรูทีนย่อยเอาไว้ตรงกลาง แล้วมีสัปรูทีนเล็กอยู่ด้านบนก็ได้ หรือ 2. จะใช้วิธีเขียนเป็นสัปรูทีนภายในสัปรูทีนซ้อน ๆ กันลงไปก็ได้ เรื่องนี้

มีทั้งดีและเสีย ถ้าเราไม่รู้จักเลือกให้เหมาะสมก็ดูจะยุ่งเหยิงเอาการ

ผมจึงได้ให้ข้อเสนอแนะว่า ถ้าเป็นโปรแกรมขนาดไม่ใหญ่นัก ให้ใช้แบบแรกแบบเดียว หรือคือ จะไม่มีการเขียนเป็นสัปรูทีนอยู่ภายในสัปรูทีนเลย สัปรูทีนทั้งหมดจึงเป็นระดับข้างเคียงหรือพี่น้องกันหมด ผมเรียกตัวล่างสุดว่าพี่ใหญ่ พี่ใหญ่สามารถเรียกใช้ตัวที่อยู่เหนือขึ้นมา ซึ่งอาจเรียกว่าเป็นพี่กลาง แต่พี่กลางนี้ก็ยังสามารถเรียกใช้สัปรูทีนที่อยู่เหนือขึ้นไปอีก ซึ่งอาจเรียกว่าน้องเล็ก ในโปรแกรมของเราจึงสามารถเขียนเป็นสัปรูทีนที่เรียกใช้กันแบบซ้อน ๆ ขึ้นมาได้ โปรแกรมในรูปที่ 1 เป็นตัวอย่างของโปรแกรมประเภทนี้ ในโปรแกรมนี้ผมมีสัปรูทีนชื่อ GETTIME, STARTTIMECOUNT, UPDATETIMECOUNT, TIMEOUT (L), DRAWBOX (X1, Y1, X2, Y2, C), BEEP, PING, PRINTSCORE (SCORE), LOADPICTURE, TURNUPTILE (I, J), TURNDOWNTILE (I, J), PUTMARK (I, J), REMOVEMARK (I, J) NEWSCREEN, DOSCRAMBLING, CHECKMATCH, CHECKTAB (SWT), CONGRATULATE, และ SOUNDEND ตามลำดับ ส่วนที่เหลือจะเป็นบอดี้ของเมนโปรแกรม

การที่ผมจัดเรียงสัปรูทีนเอาไว้ตามข้างบนนี้มีหลักการ 2 ประการ ประการแรกคือ ผมพยายามจัดเรียงตามลำดับเรื่องที่จะต้องทำ เช่น LOADPICTURE ควรมาก่อน NEWSCREEN ทั้งนี้เพราะเราจะวาดรูปตามคำสั่งใน NEWSCREEN ยังไม่ได้จนกว่าจะโหลดเอาภาพจากจานแม่เหล็กลงมาก่อน หลังจากโหลดภาพและวาดภาพเสร็จแล้ว เราก็ต้องทำ DOSCRAMBLING ซึ่งเป็นการสลับที่ของภาพต่าง ๆ ทีละคู่ ทั้งนี้เพื่อให้ผู้เล่นไม่ทราบว่าภาพไหนจะเข้าคู่กับภาพไหน



หลักเกณฑ์การวางตำแหน่งที่ของสัปรูทีนอีกข้อหนึ่งก็คือ ต้องดูว่าสัปรูทีนอันไหนจะถูกใครเรียกใช้ได้บ้าง ตัวที่จะถูกเรียกใช้นั้นต้องมาก่อนตัวผู้เรียกใช้ ยกตัวอย่างเช่น PRINTSCORE นั้นต้องมาก่อน NEWSCREEN และ CONGRATULATE ทั้งนี้เพราะมันถูกเรียกใช้โดยสัปรูทีนทั้ง 2 หรือพวก TURNUPTILE, TURNDOWNTILE, PUTMARK และ REMOVEMARK นั้นต้องมาก่อนพวก NEWSCREEN, DOSCRAMBLING, CHECKMATCH และ CHECKTAB เพราะมันเป็นรูทีนย่อย ๆ

อนึ่ง ท่านควรสังเกตว่า สัปรูทีนจำนวนหนึ่งซึ่งได้แก่ GETTIME, STARTTIMECOUNT, UPDATETIMECOUNT, TIMEOUT (L), DRAWBOX (X1, Y1, X2, Y2, C), BEEP, PING และ PRINTSCORE (SCORE) นั้นมีลักษณะเป็นสัปรูทีนทั่วไป ซึ่งจะถูกเรียกใช้ในสัปรูทีนอื่น ๆ ได้หลายแห่ง ดังนั้นเราจึงมักจะนำมาเขียนไว้ในตอนต้นของโปรแกรม สัปรูทีนพวกนี้สามารถก็อบปี้เอาไปใช้ในโปรแกรมที่คล้าย ๆ กันได้ เรื่องนี้จะช่วยร่นระยะเวลาในการพัฒนาโปรแกรมลงไปได้มาก นี่คือสิ่งที่พวกเราเรียกกันว่า "RE-USEABLE CODE" โปรแกรมเล่นเกมของผมส่วนใหญ่ต้องใช้สัปรูทีนเหล่านี้ ผมมักจะก็อบปี้มาใช้ต่อ ๆ กันไป

โปรแกรมการเล่นเกมง่าย ๆ

เอ้อ ผมต้องขอโทษท่านผู้อ่านเป็นอย่างยิ่ง คือ มัวแต่มาอธิบายถึงโครงสร้างของโปรแกรมโดยยังไม่ได้บอกเลยว่าโปรแกรมนี้ออกแบบไว้ทำอะไร โปรแกรมนี้เป็นโปรแกรมเล่นเกมง่าย ๆ อันหนึ่ง ข้อกำหนดคือ ผมจะทำภาพขนาดเล็ก ๆ เอาไว้จำนวนหนึ่ง (ในโปรแกรมนี้มี 38 ภาพ) ภาพเหล่านี้มีขนาด 23 คูณ 23 พิกเซล ภาพพวกนี้ท่านสามารถสร้างใหม่เองได้ ท่านต้องสร้างแล้วเก็บเอาไว้ในไฟล์ชื่อว่า MP1-DO1.DAT เมื่อได้แล้วท่านจะต้องโหลดภาพเหล่านี้มาเก็บไว้ที่ตัวแปร PICT ซึ่งเป็นอะเรย์ของสตริงขนาด 176 ตัวอักษร เสร็จแล้วผมจะให้โปรแกรมมันเลือก

อย่างสุ่ม (At random) ออกมา 24 ตัว ผมเอามาแสดงเป็น 3 แถว แถวละ 8 ภาพ และทำการก๊อบปี้ลงเป็นแถวที่ 4, 5, 6 ด้วย ตกลงบนจอภาพจะมี 6 แถว แถวละ 8 ภาพ แต่มีเป็นคู่กัน (แถวที่ 1 จะตรงกับแถวที่ 4 แถวที่ 2 จะตรงกับแถวที่ 5 และแถวที่ 3 จะตรงกับแถวที่ 6) การสร้างภาพเป็น 6 แถว และเป็นคู่ ๆ นี้ทำโดยสับรูทีน NEW-SCREEN

หลังจากนั้นผมก็จะให้โปรแกรมมันจัดการสลับที่ของภาพเหล่านี้ การสลับที่ทำโดยสับรูทีน DOSCRAM-BLING มันจะสลับที่อย่างเดาสุ่มเป็นจำนวน 100 ครั้ง ภาพต่าง ๆ ส่วนใหญ่จึงเปลี่ยนที่อยู่ไปหมด หลังจากนั้นแล้วเจ้า DOSCRAMBLING ก็จะคว่ำหน้าของภาพเหล่านี้จนหมด แล้วก็จะให้ท่านผู้เล่นทำการเปิดดู การเปิดดูนี้ทำได้โดยเลื่อนเครื่องหมายกากบาทไปที่ช่องของภาพที่ต้องการ (ตอนนี้จะเป็นตารางขนาด 6 แถว ๆ ละ 8 ช่อง) เสร็จแล้วกดแป้น ENTER โปรแกรมก็จะเปิดภาพนั้นให้ดูเป็นระยะเวลาประมาณครึ่งวินาที ท่านจะต้องพยายามจำเอาไว้ เสร็จแล้วท่านก็เลื่อนกากบาทไปยังช่องอื่น แล้วกด ENTER เพื่อเปิดภาพดูอีก ถ้าภาพนี้ตรงกับภาพที่แล้วมา ท่านก็จะได้คะแนน และภาพคู่นี้ก็จะเปิดหงายทิ้งเอาไว้ ท่านจะต้องทำอย่างนี้ไปเรื่อย ๆ จนกว่าจะสามารถจับคู่ได้หมด 24 คู่ เมื่อได้ครบแล้วโปรแกรมจะได้คะแนนเพิ่มอีกจำนวนหนึ่ง ทั้งนี้โดยขึ้นอยู่กับระยะเวลาที่ใช้ทั้งหมด ท่านยิ่งทำเสร็จได้เร็วก็ยิ่งจะได้คะแนนสูง ถ้าท่านใช้เวลาเกิน 600 วินาทีจะไม่ได้คะแนนแถมดังกล่าวนี้ อัตราการจ่ายจะอยู่ในสับรูทีน CONGRATULATE

ครับ เกมนี้เป็นที่ใช้ฝึกความจำได้

เป็นอย่างดี ความยากง่ายจะอยู่ที่ลักษณะของรูปภาพ และระยะเวลาที่เครื่องเปิดให้ดู รูปภาพ 38 รูปภาพที่ผมทำไว้นั้นมีความยากง่ายปานกลาง ถ้าท่านอยากจะให้ยากขึ้นอีกก็ต้องทำภาพที่ดูแล้วคล้าย ๆ กัน แต่มีรายละเอียดที่ต่างกันเล็ก ๆ น้อย ๆ การเปิดให้ดูเป็นระยะเวลาสั้นกว่าครึ่งวินาทีดูจะโหดไปหน่อย

ตัวแปรสำคัญ ๆ

จากคำอธิบายที่กล่าวมาข้างบน ผมคิดว่าท่านผู้อ่านคงจะเข้าใจดีขึ้น ท่านควรกลับไปอ่านดูลิสติงในรูปที่ 1 ใหม่อีกครั้งหนึ่ง ในโปรแกรมนี้ตัวแปรที่สำคัญ ๆ จะมีดังนี้

TAB = อะเรย์เก็บรหัสของภาพในแต่ละช่อง

TABSTA = อะเรย์เก็บสถานะของภาพ (0 = คว่ำอยู่, 1 = หงายขึ้น, และ 2 = ถูกจับคู่แล้ว)

I, J = ตำแหน่งที่ตั้งของกากบาท

I1, J1 = ตำแหน่งที่เปิดดูครั้งหลังสุด

PICT = อะเรย์เก็บภาพ 38 ภาพ

DIGI = อะเรย์เก็บภาพตัวเลข 0-9 เพื่อให้แสดงคะแนน

SCORE = คะแนน

TIMELIMIT = ระยะเวลาให้เล่นเกมทั้งหมด = 1800 วินาที

STARTTIME = เวลาเริ่มต้นแต่ละรอบ

ENDTIME = เวลาเมื่อจับคู่ได้ครบ 24 คู่

ท่านผู้อ่านควรทำความเข้าใจกับโปรแกรมนี้ให้ดี ๆ จุดสำคัญก็คือ การเลือกที่ตั้งของสับรูทีนต่าง ๆ มันจะแบ่งออกเป็น 3 กลุ่มใหญ่ ๆ คือ สับรูทีนทั่วไป (ซึ่งจะถูกเรียกใช้ที่ไหนก็ได้) สับรูทีนย่อย (ซึ่งจะถูกเรียกโดยสับรูทีนใหญ่) และตัวสับรูทีนใหญ่ แต่ทั้งหมดนี้เป็นพี่น้องกันหมด เราไม่มีตัวไหนประกาศเป็นโพรซีเดอร์ภายในโพรซีเดอร์เลย วิธีนี้เป็นวิธีที่เขียนได้ง่ายและดีบักได้ง่ายดี แต่จะเหมาะสำหรับโปรแกรมไม่ใหญ่นัก โปรแกรมเล่นเกมของผมส่วนใหญ่ก็มีขนาดเพียงเท่านี้แหละ ผมได้ลองใช้วิธีนี้มา 4-5 ปีแล้ว มันได้ผลดีเสมอมา ผมจึงเอาแนะนำให้พวกเราใช้กัน

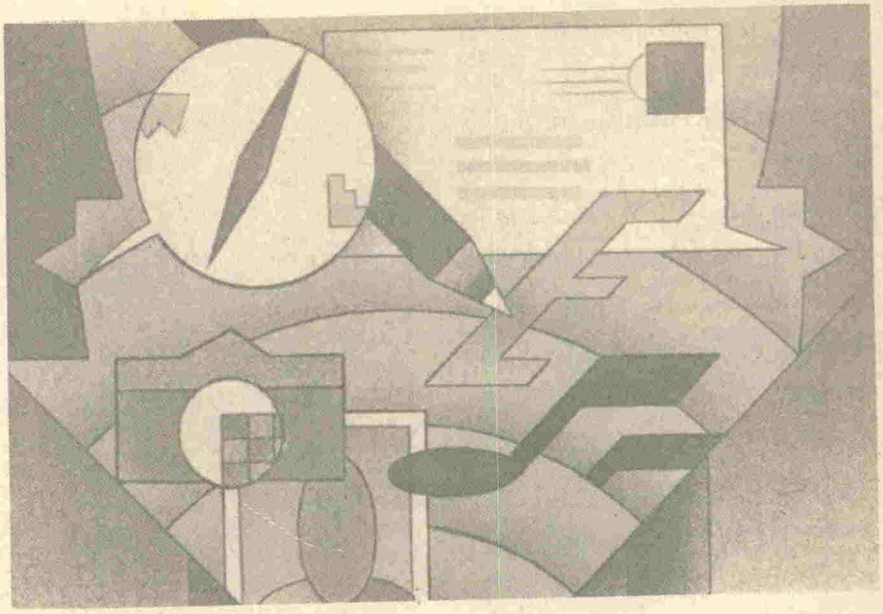
เริ่มต้นเขียนโปรแกรมและแจกจ่ายงาน

ทีนี้เรามาลองเขียนโปรแกรมแบบใช้สับรูทีนภายในสับรูทีนกันดูบ้าง เพื่อเป็นการง่ายต่อการเปรียบเทียบ ผมจะขอใช้เรื่องเดียวกัน คือ ให้เขียนเกมการจับคู่ภาพเช่นเดิม แต่มาทีนี้ผมจะแจกจ่ายงานให้ลูกทีมไปช่วยกันเขียน ผมจะเป็นผู้เขียนเมนโปรแกรมแล้วให้ลูกทีมผม 3 คนไปเขียนสับรูทีนใหญ่ 3 สับรูทีน สับรูทีนแรกผมให้เรียกชื่อว่า SETUP-GAME ผมมอบหมายให้คุณสมชายไปทำ หน้าที่ของสับรูทีนนี้คือ ให้เลือกภาพ 24 ภาพจาก PICT มาแสดงบนจอ

เป็น 24 คู่ (6 แถว ๆ ละ 8 ภาพเหมือนเดิม) แล้วทำการสลับที่ภาพเหล่านั้น 100 ครั้งเช่นกัน ทำเสร็จแล้วก็จบเพียงเท่านี้ สำหรับสัปรุทีนอันที่ 2 ผมมอบหมายให้คุณวิชัยไปทำ สัปรุทีนที่ 2 นี้ผมให้เรียกชื่อว่า INTERPRETKEY เจ้าสัปรุทีนนี้จะถูกใช้เพื่อตีความการกดแป้นคีย์บอร์ดของผู้เล่น ผู้เล่นสามารถกดแป้นต่าง ๆ ได้ดังนี้คือ

1. กด E เพื่อจบเกม
2. กดเครื่องหมายลูกศร (Arrow-up, Arrow-down, Arrow-left, Arrow-right) เพื่อเลื่อนเครื่องหมายกากบาท
3. กด ENTER เพื่อบอกให้เปิดดูภาพ

ถ้าผู้เล่นกดปุ่มอื่น ๆ จะไม่มีผลอะไร เจ้าสัปรุทีนนี้เมื่อเช็คว่าผู้เล่นกดปุ่มอะไรมันก็จะไปดำเนินการตามสั่ง เช่น ถ้ากดปุ่มตัว E ก็จะเซตค่า ENDSWT ให้เป็น 1 ทั้งนี้เพื่อบอกเมนโปรแกรมว่าผู้เล่นต้องการจบเกมแล้ว แต่ถ้ากดแป้น ENTER เจ้าสัปรุทีนนี้ก็จะต้องเช็คว่าภาพในช่องที่ (I, J) มันตรงกับช่อง (I1, J1) หรือไม่ ถ้าใช่ก็ต้องให้คะแนน และต้องเปิดภาพคู่นี้ทิ้งไว้ และต้องเซตค่า TABSTA [I, J] และ TABSTA [I1, J1] เป็นเลข 2 (TABSTA ย่อมาจาก Table-Status) เป็นต้น สัปรุทีนนี้ตีความการกดปุ่มแต่ละครั้ง เมื่อเสร็จแล้วก็ต้องจบรุทีนไปเลย มันจะได้กลับมาสู่เมนโปรแกรม ซึ่งผมจะจัดการส่งต่อไปยังสัปรุทีนที่ 3 สัปรุทีนที่ 3 นี้ผมมอบหมายให้คุณสมบูรณ์ไปทำ สัปรุทีนนี้ผมให้เรียกชื่อว่า CHECKSTATUS มันจะทำหน้าที่ตรวจดูว่ามีการจับคู่กันจนครบ 24 คู่แล้วหรือยัง ถ้ายังไม่ครบก็ผ่านไปเฉย ๆ แต่ถ้าครบแล้วก็จะต้องแสดงความยินดีกับผู้เล่น และมีการคำนวณคะแนนแถมตามจำนวนวินาทีที่



เซฟได้ เสร็จแล้วจะต้องเซตค่าของ RESTARTSWT ให้เท่ากับ 1 ทั้งนี้เพื่อบอกเมนโปรแกรมของผมให้ทราบว่าต้องขึ้นภาพชุดใหม่กันได้แล้ว

เรามีข้อตกลง

ครับ โปรแกรมขนาดใหญ่มาก ๆ นั้นเรามักจะออกแบบเค้าโครงใหญ่ ๆ เสียก่อน เสร็จแล้วมอบหมายให้คนเขาช่วยทำสัปรุทีนใหญ่ ๆ ให้เรา ผู้เขียนสัปรุทีนเหล่านี้อาจไม่ต้องรู้จักหน้าตากันเลยก็ได้ แต่ละคนสามารถใช้ชื่อตัวแปรที่ซ้ำกันได้ เจ้า "Scope Rule" จะช่วยป้องกันไม่ให้มีการไปแก้ตัวเลขของกันและกันโดยอัตโนมัติ แต่เรื่องนี้มีข้อตกลงที่สำคัญอยู่ข้อหนึ่ง คือ ห้ามใช้ตัวแปรโกลบอลโดยตรง ตัวแปรโกลบอลจะต้องถูกส่งผ่านทางพารามิเตอร์หมด สัปรุทีนใหญ่เหล่านี้จึงต้องมีลิสต์ของพารามิเตอร์ที่อยู่ในวงเล็บตามหลังมายาวเลยทีเดียว ผู้เขียนสัปรุทีนเหล่านี้จะต้องรู้จักเลือกด้วยว่าจะให้เป็นชนิดอินพุตพารามิเตอร์ หรือเอาต์พุตพารามิเตอร์ เจ้า 2 สิ่งนี้มีผลต่างกันมาก ถ้าเลือกใช้ไม่ถูกวิธีก็อาจเกิดบักขึ้นมาได้เดี๋ยวผมจะชี้ให้ดูในรายละเอียด ตอนนี้เรามาดูปัญหาของคุณสมชาย คุณวิชัย และคุณสมบูรณ์กันก่อนดีกว่า

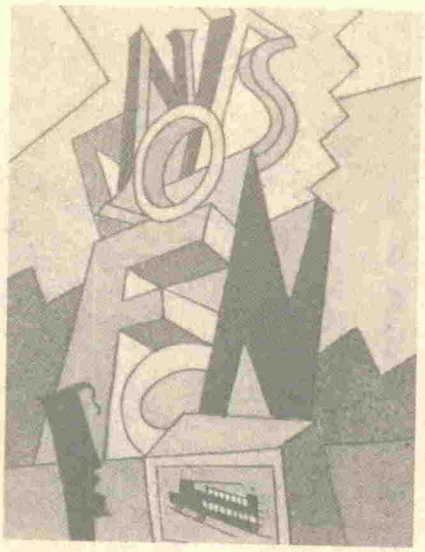
วิเคราะห์หน้าที่คุณสมชาย

เรามาเริ่มวิเคราะห์หน้าที่ของคุณสมชายกันก่อน หน้าที่หลักของสัปรุทีนนี้มี 2 ขั้นตอน ขั้นตอนแรกจะเป็นการวาดภาพ 24 คู่ (6 แถว ๆ ละ 8 ภาพ) เสียก่อน เสร็จแล้วจึงทำการสลับที่ (Scramble) ภาพเหล่านี้ เมื่อมองในแนวนี้เขาก็คิดว่า เขาควรแยกเอาเรื่องการสลับที่มาเป็นสัปรุทีนย่อย และในเมนรูทีนก็จะเป็นการเรียกใช้สัปรุทีนย่อยนี้ สำหรับเรื่องของตัวแปรต่าง ๆ ก็จะมีดังนี้คือ

– ขั้นแรกเขาต้องขอภาพ 38 ภาพจากเมนโปรแกรมเข้ามา อันนี้ทำได้โดยประกาศให้ PICT เป็นอินพุตพารามิเตอร์ ได้มาแล้วเขาก็สามารถใช้คำสั่ง PUTPIC ในภาษาปาสคาลเพื่อวาดภาพบนจอได้ การกำหนดตำแหน่งที่จะวาดภาพนี้จะต้องใช้ตัวแปร K1, K2 เพื่อบอกแถวและบอกคอลัมน์ ตัวแปรคู่นี้ใช้เป็นโลคอลได้ ตัวแปรที่สำคัญอีก 2 ตัวคือ TAB และ TABSTA เจ้าอะเรย์

คู่นี้จะถูกเซตค่าในตอนเลือกภาพจาก PICT เสร็จแล้วจะต้องถูกโยกย้ายสลับที่กันไปมา และจะต้องส่งผลออกไปสู่เมนโปรแกรม ดังนั้นตัวแปรคู่นี้ต้องประกาศเป็นเอาต์พุตพารามิเตอร์ ข้อแตกต่างระหว่างอินพุตพารามิเตอร์กับเอาต์พุตพารามิเตอร์คือ จะต้องมีคำว่า VAR อยู่หน้าชื่อพวกเอาต์พุตพารามิเตอร์ (ในตอนประกาศชื่อครั้งแรก)

หลังจากที่คุณสมชายได้คิด และได้วางแผนเค้าโครงของสัปรูทีนของเขาแล้ว เขาก็จะเขียนว่า PROCEDURE SETUPGAME (PICT: ARRAY [1...38] OF ST176; VAR TAB, TABSTA : ARRAY [1...8, 1...6] OF INTEGER); และเขียนรายละเอียดต่าง ๆ ลงไป เรื่องนี้เราจะพบว่ามันคอมพายล์ไม่ผ่าน ทั้งนี้เพราะการประกาศเป็นอะเรย์อยู่ในลิสต์ของอินพุต และเอาต์พุตพารามิเตอร์นั้นทำไม่ได้ เขาต้องกลับมาปรึกษาผม (ผู้เขียนเมนโปรแกรม) ซึ่งเรื่องนี้ผมจะต้องแก้ปัญหาให้เขา คือ ต้องเอา ARRAY [1...38] และ ARRAY [1...8, 1...6] มาประกาศเป็น TYPE AR045 และ AR 8X6 เสียก่อน เสร็จแล้วจึงให้เขาเอาไปใช้ประกาศเป็น Data Type ในสัปรูทีนของพวกเขา และในโอกาสเดียวกันนี้ผมก็ต้องบอกเขาไปด้วยว่า ผมจะทำสัปรูทีน UPDATETIMECOUNT และ PRINTSCORE (SCORE) เอาไว้ให้เขาด้วย สัปรูทีนย่อยคู่นี้จะต้องใช้ในการแสดงเวลาที่เหลืออยู่ (เป็นวินาที) บนจอภาพ และแสดงคะแนนเป็นตัวเลขขนาดใหญ่บนจอภาพ ถ้าผมไม่ทำสัปรูทีนคู่นี้ให้เขา เขาก็จะต้องเขียนเอง และจะต้องเสียเวลาประกาศให้ DIGI เป็นอินพุตพารามิเตอร์อีกด้วย การเปลี่ยนแปลงและปรับปรุงนี้ผมต้องแจ้งคุณวิชัยและคุณสมบูรณ์ด้วย



ครับ คุณสมชายได้ข้อแนะนำของผมไป เขาก็เอาไปเขียนเป็นสัปรูทีน หรือโพรซีเดอร์ดังแสดงอยู่ในรูปที่ 2 โพรซีเดอร์ใหญ่ของเขาคือ SETUPGAME (SC : INTEGER; PICT : AP045; VAR TAB, TABSTA : AR8X6);ภายในสัปรูทีนนี้เขามีสัปรูทีนย่อยชื่อ DOSCRAMBLING อยู่ด้วย

ดูปัญหาคุณวิชัย

เรามาดูปัญหาของคุณวิชัยกันบ้าง คุณวิชัยได้รับมอบหมายให้ทำสัปรูทีน INTERPRETKEY ซึ่งมีหน้าที่ตีความกดแป้นคีย์บอร์ดของผู้เล่น ในบอดี้ของสัปรูทีนนี้จะต้องสั่ง READ (KBD, CH); นี่เป็นการรับเอารหัสของแป้นคีย์บอร์ดเข้าไป CH คือ ตัวแปรชนิด CHAR ที่ใช้รองรับข้อมูลดังกล่าว เมื่อรับไปแล้วก็ใช้คำสั่ง IF เพื่อเช็คว่าเป็นตัว E หรือ ENTER หรือกลุ่มเครื่องหมายลูกศร ในกรณีที่เป็นตัว E เขาก็จะเซตค่า ENDSWT ให้เป็น 1 แล้วออกจากสัปรูทีนไป แต่ถ้าเป็นการกด ENTER เขาก็จะต้องเรียกสัปรูทีนย่อยมาใช้งาน สัปรูทีนนี้เขาให้ชื่อว่า CHECKMATCH ในสัปรูทีน CHECKMATCH นี้ถ้าพบว่ามีการจับคู่ได้ถูกต้อง (TAB [I, J] = TAB [I1, J1]) ก็จะต้องมีการสั่งหงายภาพที่ (I, J) และที่ (I1, J1) ไว้ แล้วต้องเซตให้ TABSTA [I, J] และ TABSTA [I1, J1] มีค่าเป็น 2 การสั่งเปิดและปิดภาพที่จุด (I, J) และ (I1, J1) นี้ต้องมีสัปรูทีนย่อยมารองรับอีกชั้นหนึ่ง คุณวิชัยให้ชื่อว่า TURNUPTILEIJ, TURNUPTILEI1J1 และ TURNDOWNTILEIJ ตามลำดับ ท่านผู้อ่านควรสังเกตว่า คุณวิชัยไม่ต้องมีสัปรูทีน TURNDOWNTILEI1J1 สำหรับกรณีที่ผู้เล่นกดแป้นเครื่องหมายลูกศรนั้น เราจะพบว่ารหัสที่ถูกสร้างจากคีย์บอร์ดจะมี 2 คาแรกเตอร์ ตัวแรก (=CH) จะตรงกับ CHR (27) และเราต้องสั่ง READ (KBD, CH2); เพื่อเช็คว่าตัวอักษรตัวที่ 2 เป็นอะไร ถ้าเป็น CH2 = CHR (72) มันจะตรงกับเครื่องหมายลูกศรชี้ขึ้นข้างบน ซึ่งคุณวิชัยจะต้องเลื่อนรูปกากบาทขึ้นไปข้างบน

การเลื่อนรูปกากบาทนี้คุณวิชัยทำเป็นสัปรูทีนย่อยไว้ 1 อัน ชื่อว่า PUTMARKIJ โดยสรุปแล้ว คุณวิชัยวางแผนไว้ว่าจะมีสัปรูทีนย่อยลงไปอีก 2 ชั้น สัปรูทีนย่อยเหล่านี้จะอยู่ภายใต้สัปรูทีนใหญ่ของเขา เมื่อเขียนแล้วจะแสดงอยู่ในลิสติงของโปรแกรมที่ 2 ท่านผู้อ่านควรสังเกตว่า คุณวิชัยเขียนว่า PROCEDURE INTERPRETKEY (VAR I, J, I1, J1 : INTEGER; PICT : APR045; TAB : AR8×6; VAR TABSTA : AR 8×6; VAR ENDSWT : INTEGER); นั่นแสดงว่าเขาประกาศให้ I, J, I1, J1, TABSTA และ ENDSWT เป็นเอาต์พุตพารามิเตอร์ แต่ PICT และ TAB เป็นอินพุตพารามิเตอร์ ซึ่งก็ถูกต้องดี ทั้งนี้เพราะสัปรูทีนของเขาต้องการ PICT และ TAB เข้ามาใช้งาน โดยไม่ต้องการ

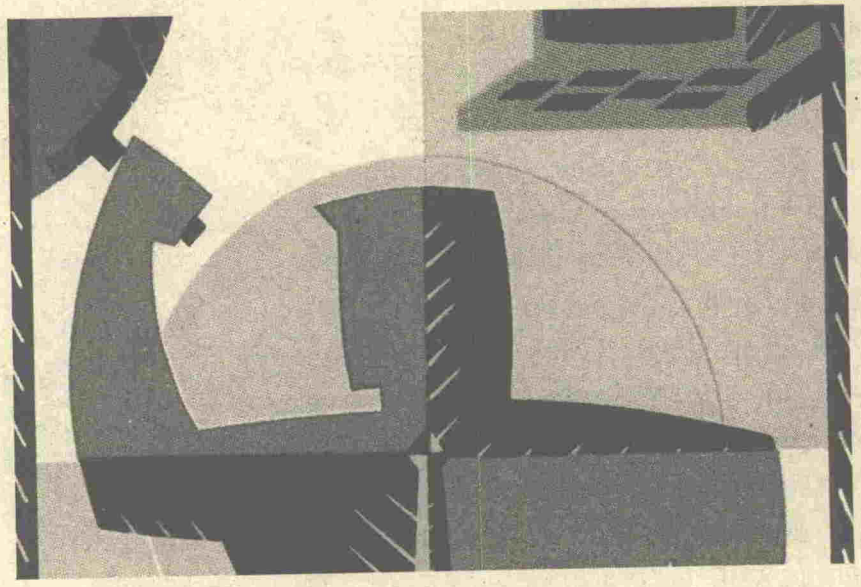
ให้มีผลกระทบออกไปภายนอก แต่ตัวแปรตัวอื่นนั้นมันต้องส่งผลกระทบออกไปภายนอกด้วย

อย่างไรก็ตาม คุณวิชัยยังทำผิดกติกาอยู่ข้อหนึ่ง คือไม่ได้ประกาศให้ SCORE เป็นเอาต์พุตพารามิเตอร์ โปรแกรมนี้หลังจากได้สัปรูทีนต่าง ๆ มารวมเป็นโปรแกรมแล้วทำงานได้ถูกต้องทุกประการ แต่ก็ยังคงผิดหลักการอยู่เล็กน้อยดังที่กล่าวมา เรื่องนี้อาจเกิดปัญหาได้เมื่อมีคนอื่นต้องมาแก้ไขโปรแกรม และอย่างน้อยเราก็จะพบว่าสัปรูทีน INTERPRETKEY นี้ ไม่สามารถจะก๊อบปี้ไปใช้ที่อื่นได้อย่างสะดวกนัก ทั้งนี้เพราะมันยังไม่สมบูรณ์ในตัวเอง (Not Completely shielded)

ดูผลงานคุณสมบูรณ์

สำหรับงานของคุณสมบูรณ์นั้นก็คล้าย ๆ ของ 2 ท่านที่กล่าวมาแล้ว คือ คุณสมบูรณ์จะต้องไปวางแผนว่าจะมีสัปรูทีนย่อยอะไรบ้าง และตัวแปรที่จะใช้ในสัปรูทีนต่าง ๆ นั้นต้องมีอะไรบ้าง และต้องเป็นชนิดไหน ผลงานของคุณสมบูรณ์เป็นดังแสดงอยู่ในรูปที่ 2 งานของคุณสมบูรณ์นี้อาจง่ายกว่าคนอื่น ๆ เล็กน้อย แต่ถ้าท่านลองอ่านดูสัปรูทีนของคุณสมบูรณ์แล้วจะพบว่าคุณสมบูรณ์ทำได้ครบถ้วนจริง ๆ เช่น ตัวแปรที่เขาต้องการใช้ ได้แก่ STARTTIME, ENDTIME, TABSTA, SCORE และ RESTARTSWT เท่านั้น และตัวที่จะเป็นเอาต์พุตพารามิเตอร์มีก็แค่ 2 ตัว คือ SCORE กับ RESTARTSWT เขาประกาศตัวแปรเหล่านี้เอาไว้ในวงเล็บหลังชื่อโพรซีเดอร์ตามลำดับความสำคัญ คือ ตัวแปรอินพุตมาก่อนประเภทเอาต์พุต

นอกจากนี้ท่านจะเห็นว่า คุณสมบูรณ์นี้ละเอียดรอบคอบมาก คือ เขาให้



ใช้ ENDTIME เป็นอินพุตพารามิเตอร์ และเวลาเมนโปรแกรมจะเรียกใช้สัปรูทีนของเขา ก็จะต้องใส่ T1 เข้าไปในคำสั่ง (ดูประโยคที่เขียนว่า CHECKSTATUS (STARTTIME, T1, TABSTA, SCORE, RESTARTSWT); ใน MAINLOOP) ที่เป็นเช่นนี้ก็เพราะค่าของ T1 ในตอนเรียกสัปรูทีน CHECKSTATUS นั้น จะเป็นเวลาที่ผู้เล่นทำการจับคู่ได้สำเร็จแล้ว (ถ้าจริง) ถ้าเขาไม่ประกาศให้ ENDTIME เป็นอินพุตพารามิเตอร์ เขาก็จะต้องไปเขียนว่า ENDTIME : = T1; ในสัปรูทีน CONGRATULATION ซึ่งเป็นการไปใช้ตัวแปรโกลบอลโดยตรง และ T1 ในตอนนี้อาจไม่ตรงกับ T1 ในตอนเริ่มต้นของสัปรูทีน CHECKSTATUS ทั้งนี้เพราะเรามีการเรียกใช้สัปรูทีน UPDATETIMECOUNT อยู่ภายในสัปรูทีนนี้ด้วย T1 ในตอนเริ่มต้นของ CHECKSTATUS จะถูกต้องและตรงกับความจริงมากกว่า

โดยสรุปแล้ว สัปรูทีนของคุณสมบูรณ์นี้มีความสมบูรณ์กว่าของคนอื่น ๆ และถ้าจะเปรียบเทียบกับของคุณวิชัยแล้วจะพบว่า คุณวิชัยเรียกใช้ตัวแปรเกินความจำเป็น คือ ในงานของคุณวิชัยนั้น เขาจะเกี่ยวข้องกับ TAB และ TABSTA ที่จุด (I, J) และ (I1, J1) เท่านั้น เขาไม่มีความจำเป็นต้องให้เมนโปรแกรมส่งค่าทุกตัวของ TAB และ TABSTA เข้าไป

สรุปบทความก่อนจาก

มาถึงตรงนี้ท่านผู้อ่านก็ได้เห็นการเขียนโปรแกรมทั้ง 2 แนวไปแล้ว แนวแรกเป็นการใช้สัปรูทีนระดับเดียวกันหมด แนวที่ 2 เป็นการยกเป็นสัปรูทีนใหญ่ 2-3 อัน แล้วในแต่ละอันจะมีสัปรูทีนย่อยภายในซ้อน ๆ กันลงไป โปรแกรมในแนวที่ 2 นั้นดูจะเขียนยากกว่า แต่เมื่อไหร่ที่เราไปเจอกับงานขนาดยักษ์ก็จะพบว่า มันให้ผลที่ดีกว่า การเขียนโปรแกรมตามแนวที่ 2 นี้จะลดตัวแปรประเภทโกลบอลลงไปได้มาก ผู้วางแผนทำเมนโปรแกรมจะคิดได้ทะลุปรุโปร่งดีกว่า โอกาสที่จะมีบักจะลดลง แต่ทีมงานทั้งทีมจะต้องเขียนอย่างมีหลักเกณฑ์ คือ ต้องระมัดระวังไม่ให้เรียกใช้ตัวแปรโกลบอลโดยตรง นี่เป็นโปรแกรมมิงเทคนิคที่ควรจะเรียนรู้เอาไว้

ต้นฉบับไม่มีหน้า 267-268

No This Page 267 to 268 In Original

รูปที่ 1 (ต่อ)

```
        BEGIN
        REMOVEMARK(I,J);
        J:=J+1;PUTMARK(I,J);
        END;
      END;
  BEEP;
  GOTO MAINLOOP;

EOJ:
  SOUNDEND;

END.
```

รูปที่ 2 โปรแกรมลิสติงของ MP 1502

```
PROGRAM MP1S02;
{$I GRAPH.P }
LABEL STARTINGPOINT,LOOP1,MAINLOOP,READKBD,EOJ1,EOJ;
CONST MAXI=8;MAXJ=6;
TYPE  ST06=STRING[6];
      ST10=STRING[10];
      ST30=STRING[30];
      ST176=ARRAY[0..176] OF BYTE;
      AR045=ARRAY[0..45] OF ST176;
      AR8X6=ARRAY[1..8,1..6] OF INTEGER;
      PICREC=RECORD
           PCS:ST06;
           PIC:ST176;
             END;
VAR CHESSPICFILE:FILE OF PICREC;
VAR CHESSPICREC:PICREC;
VAR DIGI,PICT:AR045;
VAR TAB,TABSTA:AR8X6;
VAR I,J,I1,J1,NPAL,ENDSWT,RESTARTSWT:INTEGER;
VAR SCORE,TIMELIMIT,STARTTIME,ENDTIME:INTEGER;
VAR HOUR,MIN,SEC,FRAG,T0,T1,T2,SVT,SVT1:INTEGER;

PROCEDURE GETTIME;
TYPE REGPACK=RECORD
             AX,BX,CX,DX,BP,SI,DI,DS,ES,FLAGS:INTEGER;
             END;
VAR REGS:REGPACK;
BEGIN
    WITH REGS DO
      BEGIN
      AX:=$2C00;
      MSDOS(REGS);
      HOUR:=HI(CX);
      MIN:=LO(CX);
      SEC:=HI(DX);
      FRAG:=LO(DX);
      END;
END;

PROCEDURE STARTTIMECOUNT;
BEGIN
  GETTIME;T0:=60*MIN+SEC;
  T1:=TIMELIMIT;SVT:=TIMELIMIT;
END;

PROCEDURE UPDATETIMECOUNT;
BEGIN
  GETTIME;T2:=60*MIN+SEC;
  IF T2<T0 THEN T2:=T2+3600;
  T1:=TIMELIMIT-T2+T0;
  IF T1<>SVT THEN
     BEGIN
     GOTOXY(20,3);WRITE(' TIME=',T1:4);
     SVT:=T1;
     END;
END;

PROCEDURE TIMEOUT(L:INTEGER);
VAR K1,K2:INTEGER;
BEGIN
  FOR K1:=1 TO L DO
    FOR K2:=1 TO 500 DO
      BEGIN  END;
END;

PROCEDURE BEEP;
BEGIN
  SOUND(650);TIMEOUT(10);NOSOUND;TIMEOUT(2);
END;

PROCEDURE PING;
BEGIN
  SOUND(1650);TIMEOUT(10);NOSOUND;TIMEOUT(2);
END;

PROCEDURE DRAWBOX(X1,Y1,X2,Y2,C:INTEGER);
BEGIN
  DRAW(X1,Y1,X1,Y2,C);
  DRAW(X2,Y1,X2,Y2,C);
  DRAW(X1,Y1,X2,Y1,C);
  DRAW(X1,Y2,X2,Y2,C);
END;

PROCEDURE PRINTSCORE(SC:INTEGER);
VAR K,L,PRTSWT:INTEGER;
VAR SCR:ARRAY[1..5] OF INTEGER;
BEGIN
  SCR[1]:=SC DIV 10000;SC:=SC-10000*SCR[1];
  SCR[2]:=SC DIV 1000;SC:=SC-1000*SCR[2];
  SCR[3]:=SC DIV 100;SC:=SC-100*SCR[3];
  SCR[4]:=SC DIV 10;SC:=SC-10*SCR[4];
  SCR[5]:=SC;
  PRTSWT:=0;
  FOR K:=1 TO 5 DO
    BEGIN
    L:=SCR[K];
    IF L>0 THEN PRTSWT:=1;
    IF PRTSWT=1 THEN
      PUTPIC(DIGI[L],24*(K-1),23);
    END;
END;

PROCEDURE LOADPICTURE;
VAR K:INTEGER;
BEGIN
  ASSIGN(CHESSPICFILE,'MP1D01.DAT');
  RESET(CHESSPICFILE);
  CLEARSCREEN;
  DRAWBOX(0,0,24,24,1);
  FILLSHAPE(1,1,2,1);
  GETPIC(PICT[0],1,1,23,23);
  FOR K:=0 TO 9 DO
    BEGIN
    SEEK(CHESSPICFILE,K+2);
    READ(CHESSPICFILE,CHESSPICREC);
    DIGI[K]:=CHESSPICREC.PIC;
    END;
  FOR K:=1 TO 38 DO
    BEGIN
    SEEK(CHESSPICFILE,K+13);
```

รูปที่ 2 (ต่อ)

```
    READ(CHESSPICFILE,CHESSPICREC);
    PICT[K]:=CHESSPICREC.PIC;
    END;
  CLOSE(CHESSPICFILE);
END;


PROCEDURE SETUPGAME(SC:INTEGER;PICT:AR045;VAR TAB,TABSTA:AR8X6);
VAR K,SHIFT,K1,K2:INTEGER;
VAR USEDITEM:ARRAY[1..45] OF INTEGER;
  PROCEDURE DOSCRAMBLING;
  VAR K,K1,K2,X1,Y1,X2,Y2,T1,T2:INTEGER;
  BEGIN
    FOR K:=1 TO 100 DO
      BEGIN
      REPEAT
        X1:=RANDOM(8)+1;
        Y1:=RANDOM(6)+1;
        X2:=RANDOM(8)+1;
        Y2:=RANDOM(6)+1;
        UNTIL (X1<>X2) OR (Y1<>Y2);
      T1:=TAB[X1,Y1];
      T2:=TAB[X2,Y2];
      TAB[X1,Y1]:=T2;
      TAB[X2,Y2]:=T1;
      PUTPIC(PICT[TAB[X1,Y1]],60+24*(X1-1),60+24*(Y1-1));
      PUTPIC(PICT[TAB[X2,Y2]],60+24*(X2-1),60+24*(Y2-1));
      PING;TIMEOUT(20);
      END;
    FOR K1:=1 TO 8 DO
     FOR K2:=1 TO 6 DO
      BEGIN
      PUTPIC(PICT[0],60+24*(K1-1),60+24*(K2-1));
      BEEP;TIMEOUT(100);
      END;
  END;
BEGIN
  CLEARSCREEN;
  DRAW(0,24,0,199,1);
  DRAW(319,24,319,199,1);
  DRAW(0,24,319,24,1);
  DRAW(0,199,319,199,1);
  PRINTSCORE(SC);
  GOTOXY(20,1);WRITE('MATCHING PICTURES');
  GOTOXY(20,2);WRITE('  BY DR.PUNYA    ');
  FOR K:=1 TO 45 DO
    USEDITEM[K]:=0;
  FOR K1:=1 TO 8 DO
   FOR K2:=1 TO 3 DO
    BEGIN
    REPEAT
      SHIFT:=RANDOM(38);
      UNTIL USEDITEM[SHIFT]=0;
    TAB[K1,K2]:=SHIFT;
    TAB[K1,K2+3]:=TAB[K1,K2];
    USEDITEM[SHIFT]:=1;
    END;
  FOR K1:=1 TO 8 DO
   FOR K2:=1 TO 3 DO
    BEGIN
    TABSTA[K1,K2]:=0;
    PUTPIC(PICT[TAB[K1,K2]],60+24*(K1-1),60+24*(K2-1));
    TABSTA[K1,K2+3]:=0;
    PUTPIC(PICT[TAB[K1,K2+3]],60+24*(K1-1),60+24*(K2+3-1));
    END;
  DOSCRAMBLING;
  DRAW(60+5,60-11,60+17,60-11,0);
  DRAW(60+11,60-5,60+11,60-17,0);
END;

PROCEDURE INTERPRETKEY(VAR I,J,I1,J1:INTEGER;PICT:AR045;
          TAB:AR8X6;VAR TABSTA:AR8X6;VAR ENDSWT:INTEGER);
LABEL E1;
VAR CH,CH1,CH2:CHAR;
  PROCEDURE PUTMARK(I,J:INTEGER);
  BEGIN
    DRAW(60+24*(I-1)+5,60+24*(J-1)-11,60+24*(I-1)+17,60+24*(J-1)-11,0);
    DRAW(60+24*(I-1)+11,60+24*(J-1)-5,60+24*(I-1)+11,60+24*(J-1)-17,0);
  END;
  PROCEDURE REMOVEMARK(I,J:INTEGER);
  BEGIN
    IF (TABSTA[I,J]=2) OR (TABSTA[I,J]=1) THEN
      PUTPIC(PICT[TAB[I,J]],60+24*(I-1),60+24*(J-1));
    IF TABSTA[I,J]=0 THEN
      PUTPIC(PICT[0],60+24*(I-1),60+24*(J-1));
  END;
  PROCEDURE CHECKMATCH(I,J:INTEGER;VAR I1,J1:INTEGER;
       PICT0,PICTIJ,PICTI1J1:ST176;TABIJ,TABI1J1:INTEGER;
       VAR TABSTAIJ,TABSTAI1J1:INTEGER);
    PROCEDURE TURNUPTILEIJ;
    BEGIN
      PUTPIC(PICTIJ,60+24*(I-1),60+24*(J-1));
      IF TABSTAIJ=0 THEN TABSTAIJ:=1;
    END;
    PROCEDURE TURNUPTILEI1J1;
    BEGIN
      PUTPIC(PICTI1J1,60+24*(I1-1),60+24*(J1-1));
      IF TABSTAI1J1=0 THEN TABSTAI1J1:=1;
    END;
    PROCEDURE TURNDOWNTILEIJ;
    BEGIN
      PUTPIC(PICT0,60+24*(I-1),60+24*(J-1));
      IF TABSTAIJ=1 THEN TABSTAIJ:=0;
    END;
    PROCEDURE PUTMARKIJ;
    BEGIN
      DRAW(60+24*(I-1)+5,60+24*(J-1)-11,60+24*(I-1)+17,60+24*(J-1)-11,0);
      DRAW(60+24*(I-1)+11,60+24*(J-1)-5,60+24*(I-1)+11,60+24*(J-1)-17,0);
    END;
  BEGIN
    IF TABSTAIJ=0 THEN
      BEGIN
      TURNUPTILEIJ;
      TIMEOUT(500);
      TURNDOWNTILEIJ;
      IF ((I1<>I) OR (J1<>J)) AND (TABI1J1=TABIJ) THEN
        BEGIN
        TURNUPTILEI1J1;
        PING;TIMEOUT(20);
        TURNUPTILEIJ;
```

ต้นฉบับไม่มีหน้า 271

No This Page 271 In Original